Многоуровневый параллельный алгоритм k-декомпозиции гиперграфов

Н.В. Старостин, А.В. Филимонов

Нижегородский государственный университет им.Н.И.Лобачевского
nvstar@iani.unn.ru, andrey.v.filimonov@gmail.com

Постановка задачи

Декомпозиция гиперграфов – важная проблема, находящая широкое прикладное применение во многих областях, включая научные вычисления, проектирование радиоэлектронной аппаратуры, геоинформационные системы и исследование операций. Задача k-декомпозиции состоит в распределении вершин взвешенного гиперграфа $H(V,E,w)$[1] по k подмножествам таким образом, чтобы вес сечения, т.е. суммарный вес гиперребер, «соединяющих» вершины различных подмножеств был минимален. Формальная постановка рассматриваемой проблемы выглядит следующим образом:

Распределение предполагает, что множества $V_1,...,V_k$ попарно не пересекаются и при объединении составляют исходное множество V:

$$V_i \cap V_j = \varnothing,\ \ i,j = \overline{1,k},\ \ i \neq j,\ \ \bigcup_{i=1}^{k} V_i = V. \qquad (1)$$

Подмножества разбиения определяют подграфы разбиения $H_i(V_i,E_i)$. Значение k фиксировано и задается как параметр задачи.

Определим декомпозиционные ограничения так, чтобы вес каждого подграфа разбиения, лежал в следующих пределах:

$$L_i \le w(V_i) \le U_i, i = \overline{1,k}, \qquad (2)$$

где $w(V_i) = \sum_{v_j \in V_i} w(v_j)$, $L_i, U_i \in R$, $i = \overline{1,k}$.

В ограничениях (2) константы L_i и U_i определяют пределы, в которых могут варьироваться веса подграфов разбиения.

Вес сечения определяется суммой весов ребер, которые целиком не принадлежат ни одному подграфу разбиения:

$$W_c = \sum_{e_c \in E_c} w(e_c) \to min, \qquad (3)$$

где $E_c = \left\{ e \in E \mid \exists v_i, v_j \in e : v_i \in V_a, v_j \in V_b, a \neq b, a,b = \overline{1,k} \right\}$

Задачу (1)-(3) будем называть задачей *k*-разбиения или *k*-декомпозиции гиперграфа.

Многоуровневая парадигма

Поставленная задача NP-трудна, что обуславливает интерес к различного рода эвристическим алгоритмам поиска приближенного решения хорошего качества за приемлемое время. Одной из парадигм организации такого рода эвристик является так называемая многоуровневая схема алгоритма. Схема предполагает решение ряда задач, получающихся из исходной путем удаления несущественной с точки зрения качества получаемого решения информации. Решение таких задач меньшей размерности позволяет построить решение исходной задачи. Приложение этой парадигмы к декомпозиционным алгоритмам на гиперграфах направлено в основном на решение задач больших размерностей. Так, существующая реализация одного из таких алгоритмов [1,2,3], соз-

[1] Здесь $w : E \to R$ – отображение, определяющее вес каждого гиперребра

данная авторами позволяет, используя персональный компьютер средней конфигурации, строить разбиение гиперграфов размерностей в сотни тысяч вершин. Однако, формализация прикладных задач порождает гиперграфы, размерности которых исчисляются десятками и сотнями миллионов. Решение такого рода задач затруднительно без использования параллельных вычислителей, поскольку неминуемо приводит к дефициту памяти и вычислительного ресурса. Таким образом, разработка параллельного декомпозиционного алгоритма – актуальная научная проблема с большим прикладным значением.

Ключевая идея многоуровневого алгоритма декомпозиции состоит следующем: вместо того, чтобы строить k-разбиение непосредственно для исходного гиперграфа, сначала строится ряд его приближений (загрублений). Каждое загрубление уменьшает (редуцирует) размерность исходной задачи. Процесс редукции продолжается до тех пор, пока порядок гиперграфа не снизится до сотен или даже десятков. На гиперграфе таких размерностей и отыскивается так называемое начальное разбиение. Полученное решение используется для построения решения для исходной задачи. Это достигается путем выполнения серии переносов решения задачи меньшей размерности на задачу большей размерности с последующим улучшением перенесенного решения. Таким образом, работу многоуровневого алгоритма можно разделить на три фазы: первая – фаза загрубления, когда производится ряд последовательных редукций размерности задачи, вторая – фаза поиска начального разбиения и третья – фаза восстановления решения, когда производится серия последовательных переносов решения задачи меньшей размерности на менее редуцированную задачу с последующим улучшением спроецированного решения.

Многоуровневый параллельный алгоритм

Построение параллельного алгоритма предполагает распределение информации об исходном гиперграфе и его загрублениях между вычислительными ядрами. Это, в свою очередь, порождает ряд проблем, связанных с обеспечением параллельной обработки различных частей гиперграфа. Распределение информации о гиперграфе, т.е. назначение каждой вершине вычислителя, которым она будет обрабатываться, означает, что в общем случае будут существовать гиперребра, инцидентные вершинам, принадлежащим разным вычислителям. Наличие такого рода гиперребер, называемых «внешними», обуславливает необходимость глобальных коммуникаций барьерного характера в ходе работы многоуровневого алгоритма, что негативно сказывается на общем времени его выполнения и загрузке вычислительных узлов. Рассмотрим проблемы, возникающие при работе алгоритма подробнее.

Прежде всего, построение параллельного алгоритма декомпозиции требует рационального распределения информации о гиперграфе $H(V,E)$ по p процессорам. Предлагается использовать алгоритм region growing для формирования подграфов, принадлежащих каждому вычислителю. Алгоритм работает следующим образом. Определяется набор вершин, принадлежащих первому процессору. Для этого выбирается произвольная вершина и помечается как принадлежащая процессору 1. Таким же образом помечаются все инцидентные ей вершины, затем вершины, инцидентные им, и так далее, пока количество помеченных вершин не достигнет величины $|V|/p$. Затем из числа непомеченных вершин, «неуместившихся» на процессор 1 выбирается произвольная вершина, и процедура повторяется для процессора 2. Аналогично формируются списки вершин для остальных процессоров.

Фаза загрубления предполагает уменьшение размерности задачи за счет сокращения количества вершин и ребер гиперграфа. Предлагается использовать парное связывание вершин, которое работает следующим образом. Случайно выбирается вершина,

из числа инцидентных ей вершин выбирается та, для которой максимальна метрика связности. Метрика связности для двух вершин $\{u,v\}$ есть суммарный вес их общих инцидентных ребер. Две вершины маркируются, как связанные между собой. Затем процедура повторяется для немаркированных вершин: случайно выбирается немаркированная вершина и из немаркированных соседей назначается пара. Параллельная формулировка этого метода предполагает, что выбор пар будет производиться одновременно несколькими процессорами, а следовательно, будут возникать конфликтные ситуации, связанные с маркированием вершин, принадлежащих разным процессорам. Разрешение каждого такого конфликта ведет к тому, что количество точечных коммуникаций типа процессор-процессор сильно возрастает, что негативно сказывается на производительности алгоритма при работе на системах, основанных на передаче сообщений. Предлагается использовать двухшажную схему вычисления пар. На первом шаге все процессоры вычисляют пары, обрабатывая, в том числе, смежные вершины, принадлежащие другим процессорам. Далее процессор выполняет процедуру замены каждой пары внутренних вершин одной вершиной. Никакой коммуникации между процессорами на данном шаге не требуется. Второй шаг заключается в разрешении конфликтов в назначении вершин, связанных с внешними, т.е. принадлежащими другим процессорам, вершинами. Процессор формирует ряд пакетов с информацией о своих назначениях для внешних вершин и рассылает их процессорам-владельцам этих внешних вершин. Процессор-владелец, обладая такой информацией, может вычислить, присутствует ли конфликт. После вычисления конфликтов для всех вершин, процессор-владелец отсылает назад информацию о разрешении этих конфликтов. Эта информация будет использована на следующей итерации алгоритма загрубления для назначения пар. Таким образом, внешние вершины назначаются на следующей итерации фазы загрубления и не оказывают влияния на выполнение текущей итерации. Асинхронный характер обмена информацией о конфликтах между процессорами и отсутствие барьерных синхронизаций позволяет избежать появления узких мест в алгоритме, обусловленных межпроцессорным обменом.

Фаза поиска начального разбиения не требует параллельной формулировки, поскольку оперирует гиперграфами сравнительно малой размерности, исчисляемой сотнями вершин. Разбиение такого гиперграфа может быть выполнено последовательным алгоритмом.

Для организации улучшения решения в ходе фазы восстановления предлагается параллельный аналог жадного алгоритма перемещения вершин из одного подграфа разбиения в другой. Последовательный алгоритм работает следующим образом. Случайно выбирается вершина и перемещается в другой подграф, так, что не нарушаются ограничения балансировки и перемещение дает максимальное уменьшение веса сечения. Параллельная формулировка этого алгоритма предполагает изменение порядка, в котором обходятся вершины. В гиперграфе выделяются независимые множества вершин путем построения правильной раскраски «жадным» алгоритмом. Затем выполняется c фаз улучшения, где c – количество цветов, полученных при решении задачи раскраски. На i-той фазе рассматриваются только вершины i-того цвета. Формируется набор вершин, перемещение которых приводит к максимальному уменьшению веса сечения. Поскольку все вершины одного цвета принадлежат независимому множеству, гарантируется, что общее уменьшение веса сечения при групповом переносе равно сумме выигрышей от перемещения вершин друг за другом. После переноса обновляются внешние степени всех вершин, смежных с участвующими в перемещении и алгоритм переходит к рассмотрению вершин цвета $i+1$. Эта последовательность шагов повторяется до тех пор, пока алгоритм способен улучшить вес сечения. Балансные ограничения в этой схеме поддерживаются следующим образом. Изначально каждый про-

цессор имеет информацию о весе каждого подграфа разбиения. В ходе перемещения вершин эта информация используется для соблюдения ограничений баланса. При перемещении вершины, процессор локально обновляет информацию о весах подграфов. В конце каждой фазы перемещения веса каждого подграфа перевычисляются глобально.

Описанная выше схема организации многоуровневого параллельного алгоритма обладает несколько худшими характеристиками в терминах качества получаемого решения по сравнению с последовательным аналогом, описанным в [1]. Это обусловлено тем, что в ходе фаз загрубления и восстановления процессоры оперируют локальными данными и не имеют глобального видения процесса решения. Это естественно сказывается на качестве получаемого решения, но в то же время позволяет увеличить степень параллельности обработки гиперграфа.

Литература

1. Д.И. Батищев, Н.В. Старостин, А.В. Филимонов. Многоуровневая декомпозиция гиперграфовых структур. Приложение к журналу «Информационные технологии» №5 (141) 2008, стр.1 – 32
2. Д.И. Батищев, Н.В. Старостин, А.В. Филимонов. Многоуровневый генетический алгоритм решения задачи декомпозиции гиперграфа. Известия СПбГЭТУ "ЛЭТИ". Вып. 2/2007. Серия Информатика, управление и компьютерные технологии", Санкт-Петербург, изд-во СПбГЭТУ "ЛЭТИ". 2007 г., стр. 3-13.
3. Н.В.Старостин, А.В. Филимонов. Аспекты программной реализации гиперграфов. Труды Нижегородского государственного технического университета. Серия «Информационные технологии», Том 56, Нижний Новгород, 2005, стр. 80-89.